## REMARKS

Claims 1-37 are pending in this Application. Claim 13 has been amended. Applicant gratefully acknowledges the Examiner's indication of allowable subject matter in claims 17-25 and 31, as well as in claims 5-9, 13-16, 30, 35, and 36. The application has been carefully reviewed in light of the Office Action mailed on June 29, 2004. Reconsideration of all outstanding objections and rejections in light of the above amendments and following remarks is respectfully requested.

Claim 13 has been objected to for informalities. Claim 13 has been amended to recite "... a circuit of claim [[11]] 12 further comprising ..." as requested by the Examiner. Accordingly, Applicant requests withdrawal of the objection to claim 13.

Claims 1-4, 10-12, 26-29, 32-34 and 37 stand rejected under 35 U.S.C. § 102(e) as being anticipated by Parham (U.S. Patent No. 5,805,860). In particular, the Office Action on page 3, paragraph 6, states that Parham discloses "a computer implemented method and system for extracting flat data from a hierarchical representation of a circuit ... comprising ... a processing sequence which if flat data has been previously stored, appends previously stored data to an accumulated flat data ..." citing FIGS. 11-25 of Parham.

Parham, FIGS. 11-25, does not disclose, *inter alia*, "a processing sequence that selects an element ... a processing sequence which, if flat data has been previously stored, appends previously stored flat data to an accumulated flat data ...", as in claim 1. Unlike claim 1, Parham discloses that if a hierarchical element, e.g., instance, net or port was already stored, then processing terminates or skips storing steps. Parham does not disclose appending previously stored flat data so that each hierarchical element in a netlist does not have to be parsed and stored, but rather can be determined to have been previously stored and appended to an accumulated flat data based on previously stored flat data. For example, FIG. 11 does not disclose, *inter alia*, "a processing sequence which, if flat data has been previously stored, appends previously stored flat data to an accumulated flat data

..." as in claim 1. Rather, at most FIG. 11 discloses storing data for top level modules that have not been previously stored (e.g., processing segment 306, 308).

Another example, FIG. 12 shows that each instance (step 353), port (step 354) and net (step 355) in the hierarchical netlist is selected and "H point" is created for a selected element in a hierarchical element. FIG. 12, step 358 discloses that every instance in each module is parsed until a bottom instance is reached in the hierarchical netlist (step 362), when processing terminates.

FIG. 13 describes creating an H-point for each selected hierarchical element without "append[ing] previously stored flat data to an accumulated flat data...", as in claim 1. FIG. 14, step 410, skips processing steps if data structures for a selected instance has already been created.

In FIG. 16, at steps 452 and 462, processing is terminated or processing steps are skipped without storing data if a data structure for a selected instance has been created. In FIG. 17, step 482 and FIG. 18, step 492, processing is terminated or processing steps are skipped without storing flat data if a data structure for a selected instance has been created.

In FIG. 19, there is no disclosure of "append[ing] previously stored flat data to an accumulated flat data...", as in claim 1. Instead, each top, bottom and leaf segment is parsed and data stored. FIG. 20 discloses parsing every instance and storing data each time previously unselected instance is encountered without "append[ing] previously stored flat data to an accumulated flat data...." Processing in FIGS. 21 and 22 occurs when unselected instances or ports are encountered, only selects "unprocessed" instances (Step 624) or ports (Step 644) and only "adds appropriate elements [or ports] to list" if "not already in list" (Step 628; Step 648). In FIG. 23, there is no disclosure of "append[ing] previously stored flat data to an accumulated flat data....", such as in claim 1 (e.g., steps 674 and 678 skips processing if element or port "not already in list") FIG. 24 discloses parsing each port or element and storing parsed next/previous elements without "append[ing] previously stored flat data to an accumulated flat data…." FIG. 25 also does not discloses "append[ing] previously stored flat data to an accumulated flat data…" as in claim 1.

Accordingly, claim 1 is allowable over Parham.

Claims 2-4 and 10-11 depend directly or indirectly from claim 1, thus are allowable along with claim 1 and for other reasons. For example, there is no disclosure in FIGS. 16-25 of, *inter alia*, "execution of processing sequences for selection of an element, determination of previous storage of the element and appending of stored flat data is performed recursively" as in claim 2. Thus, claims 2-4 and 10-11 are allowable over Parham.

With respect to claim 12, the Office Action on page 3, paragraph 8 states "Parham discloses a processing sequence for determining if a selected cell instance has a matching flat path segment stored at FIGS. 21-23 and a processing sequence for retrieving flat data for the cells at FIG. 25."

Claim 12 recites, *inter alia*, "… a processing sequence for determining if a selected cell instance has a matching flat path data segment stored within the first data structure …." Unlike claim 12, Parham's FIGS. 21 and 22 discloses Parham's system "adds appropriate elements [or ports] to list" if "not already in list" (Step 628; Step 648). In FIG. 23, hierarchical elements of a hierarchical netlist is parsed without a determination if a selected cell instance in has a matching flat path segment stored. Instead, Parham only determines if it has encountered a particular instance without regard to "a matching flat path segment stored" as in claim 12. In FIG. 23, at steps 674 and 678, processing steps are skipped unless a hierarchical element or port is "not already in list". Thus, Parham would not "retreiv[e] the flat path data segment for the matching cell instance" given that processing is predicated on an element "not already in list".

Claim 12 further recites, *inter alia*, " ... a processing sequence for retrieving the flat path data segment for the matching cell instance ...." Unlike claim 12, Parham does not disclose, *inter alia*, "retrieving the flat path data segment for the matching cell instance ...." Instead, hierarchical data is parsed from the hierarchical netlist data, unlike claim 12 where "flat path data segment for the matching cell instance" is retrieved. Accordingly, claim 12 is allowable over Parham.

The Office Action states on page 4, paragraph 9 with respect to claim 26 that Parham discloses a means for identifying a repetitive cell instance elements of the hierarchical representation at COL. 20, lines 65 to COL. 21, line 12.

Claim 26 recites, *inter alia*, "a means for identifying repetitive cell instance elements of the hierarchical representation of a circuit and storing cell element instance identifiers in a first data structure ...." In contrast, Parham at COL. 20, lines 66 to line 67 states "step 353 may perform method 370 for each instance present in the given module." Parham at COL. 21, lines 1-12 indicates that a request to create an H point is received for a specific type of netlist element, then memory is allocated and assigned to a specific H point. Then, the H point type is stored into the allocated memory and a pointer to the netlist associated with a selected element of the netlist is stored in the allocated memory. Nowhere in the cited section of Parham is disclosed, *inter alia*, "identifying repetitive cell instance elements of the hierarchical representation of a circuit and storing cell element instance identifiers in a first data structure ..." as in claim 26.

The Office Action further states with respect to claim 26 that Parham discloses "a means for retrieving the flat data segments for the selected element and appending the retrieved flat data segment" at FIGS. 115 (sic) through 25." Parham does not disclose, *inter alia*, "retrieving the flat data segments for the selected element and appending the retrieved flat data segment ..." For reasons including those discussed above, Parham does not "retriev[e] the flat data segments for a selected element and append the retrieved flat data segment. Accordingly, claim 26 is allowable over Parham.

Claim 27 depends from claim 26, thus is allowable along with claim 26 and for other reasons. For example, claim 27 recites, *inter alia*, “a means for masking specified flat data being input into the second data structure.” The Office Action states it relies upon the same rejections as for claim 3 in rejecting claim 27. The Office Action generically rejects claim 3 based upon unspecified parts of Parham’s FIGS. 16-25. However, there is no disclosure in any of FIGS. 16-25 of “masking of specified flat data”, much less “masking specified flat data being input into [a] second data structure.” Accordingly, claim 27 is allowable over Parham.

The Office Action rejects claims 28 and 29 using the same grounds as used in claim 1. Claim 28 recites, *inter alia*, “if flat data has been previously stored, then appending previously stored flat data to an accumulated flat data that is traversally related to the selected element.” Claim 29 recites, *inter alia*, “if flat data has been previously stored, then appending previously stored flat data to an accumulated flat data that is traversally related to the selected element.” For at least the same reasons as indicated with respect to claim 1 above, Parham does not disclose “appending previously stored flat data to an accumulated flat data ....” Accordingly, both claim 28 and 29 are allowable over Parham.

The Office Action on page 5, paragraph 12 states with respect to claim 32 that Parham discloses “retrieving a stored sequence of flat data segment, assembling the flat data segment and storing the flat data segment” citing “Figures cited above”. Paragraph 12 of the Office Action refers to FIGS. 15-25 with respect to one of the other elements of claim 32. Accordingly, for the purposes of this response, Applicant will assume FIGS. 15-25 are being relied upon in rejecting this claim element.

Claim 32 recites, *inter alia*, “retrieving a stored sequence of flat data segments which describe flat data for higher level instances in the traversal sequence; assembling the flat data segment with the retrieved sequence of flat data segments; and storing the assembled segments in a second data structure.” However, the cited figures of Parham do not disclose, *inter alia*, “retrieving a stored sequence of flat data segments which describe flat data for higher level instances in the traversal sequence" or "assembling the flat data segment with the retrieved sequence of flat data segments". Parham also does not disclose, *inter alia*, "and storing the assembled segments in a second data structure." Rather, Parham selects each and every instance in a hierarchical netlist and parses the instance then instance data is stored as discussed above. Accordingly, claim 32 is allowable over Parham.

The Office Action on page 5, paragraph 13, states that Parham at FIGS. 23-25 teaches all elements of claim 33. However, claim 33 is allowable over Parham as the Office Action fails to cite all elements of claim 33 and for other reasons. For example, claim 33 recites, *inter alia*, " ... if the selected first element is a cell, storing an identifier for the first element into a first data structure ... storing flat data describing flat path data comprised of element instance identifiers for elements within the first element to a second data structure; and storing flat data describing flat path data comprised of element instance identifiers within the first element within a third data structure ...." The cited figures of Parham do not disclose, *inter alia*, the recited "first data structure ... second data structure ...and ... third data structure ...." Accordingly, claim 33 is allowable over Parham.

Claim 34 depends from claim 33, thus is allowable along with claim 33, and for other reasons.

The Office Action on page 6, paragraph 15 states with respect to claim 37 that Parham discloses "determining if the first element has the same flat data as a previously selected element, and if it is retrieving the stored flat data" citing Parham, Abstract and COL. 3, lines 6-50. The Office Action continues by stating that Parham further discloses "combining and storing a copy of the retrieved flat data path segment with a cumulatively combined flat data containing a sequence of flat data path segments" citing "above" and FIGS. 18-22.

Claim 37 recites, *inter alia*, "determining if the first element has the same flat data as a previously selected element, wherein said previously selected element's flat path data segment has been previously determined and stored in a first data structure ...."

Parham does not disclose, for example for the reasons discussed above, "said previously selected element's flat path data segment has been previously determined and stored in a first data structure ...." However, in Parham each selected instance is parsed then instance data is stored as discussed above. Moreover, Parham does not disclose, *inter alia*, "combining a copy of the retrieved flat path data segment with a cumulatively combined flat data containing a sequence of flat path data segments stored in a second data structure [and] ... storing said cumulatively combined flat data in the second data structure" as recited in claim 37. Accordingly, claim 37 is allowable over Parham.

In view of the above, each of the presently pending claims in this application is believed to be in immediate condition for allowance. Accordingly, the Examiner is respectfully requested to pass this application to issue.

Dated: September 29, 2004

Respectfully submitted,

By [illegible]
Thomas J. D'Amico
Registration No.: 28,371
Christopher A. Monsey
Registration No.: 53,342
DICKSTEIN SHAPIRO MORIN & OSHINSKY LLP
2101 L Street NW
Washington, DC 20037-1526
(202) 785-9700
Attorneys for Applicant